FIG.1

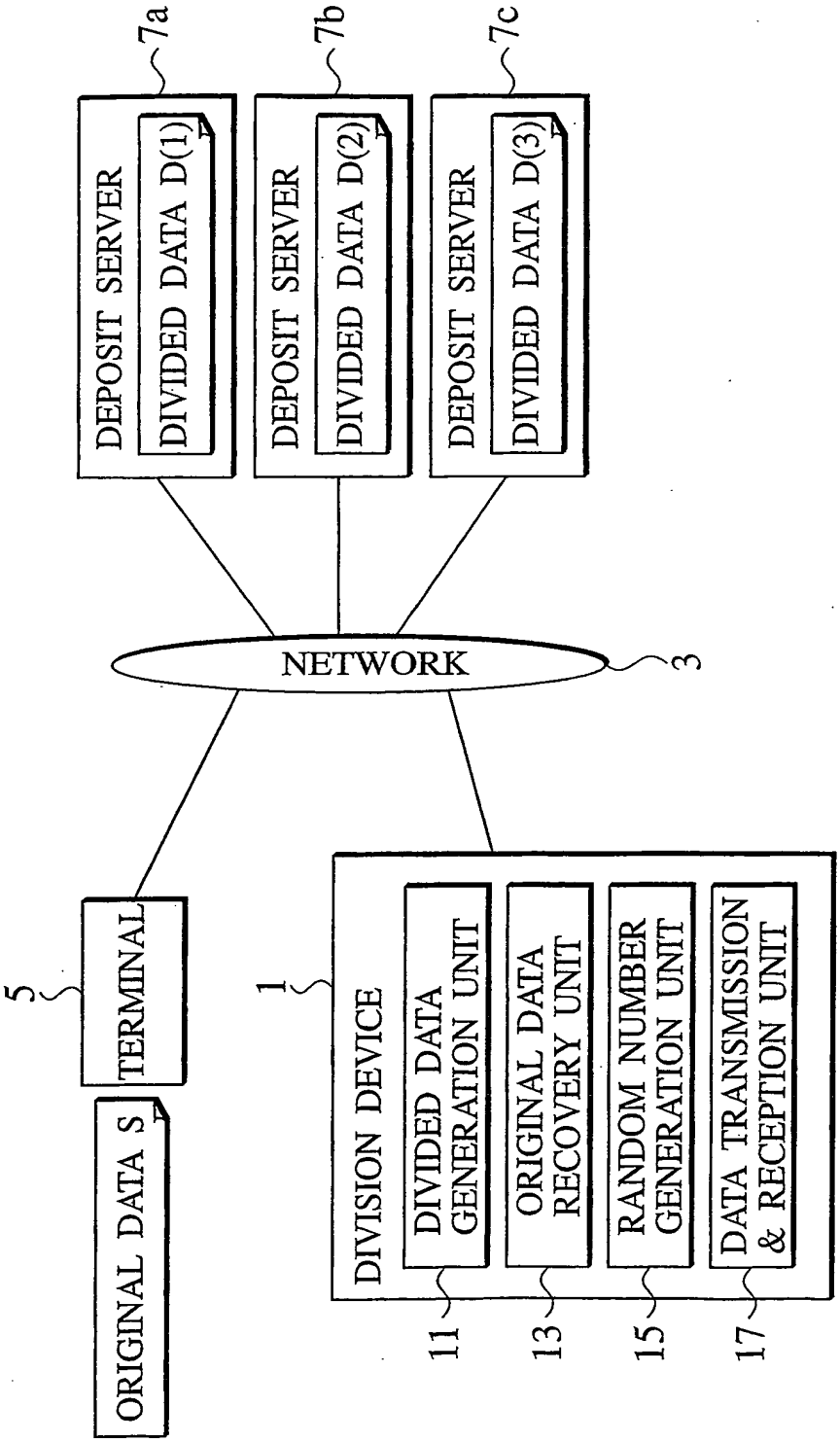
7a
7b
7c
DEPOSIT SERVER
DIVIDED DATA D(1)
DEPOSIT SERVER
DIVIDED DATA D(2)
DEPOSIT SERVER
DIVIDED DATA D(3)
NETWORK
3
5
TERMINAL
ORIGINAL DATA S
1
DIVISION DEVICE
DIVIDED DATA GENERATION UNIT
ORIGINAL DATA RECOVERY UNIT
RANDOM NUMBER GENERATION UNIT
DATA TRANSMISSION & RECEPTION UNIT
11
13
15
17

FIG.2

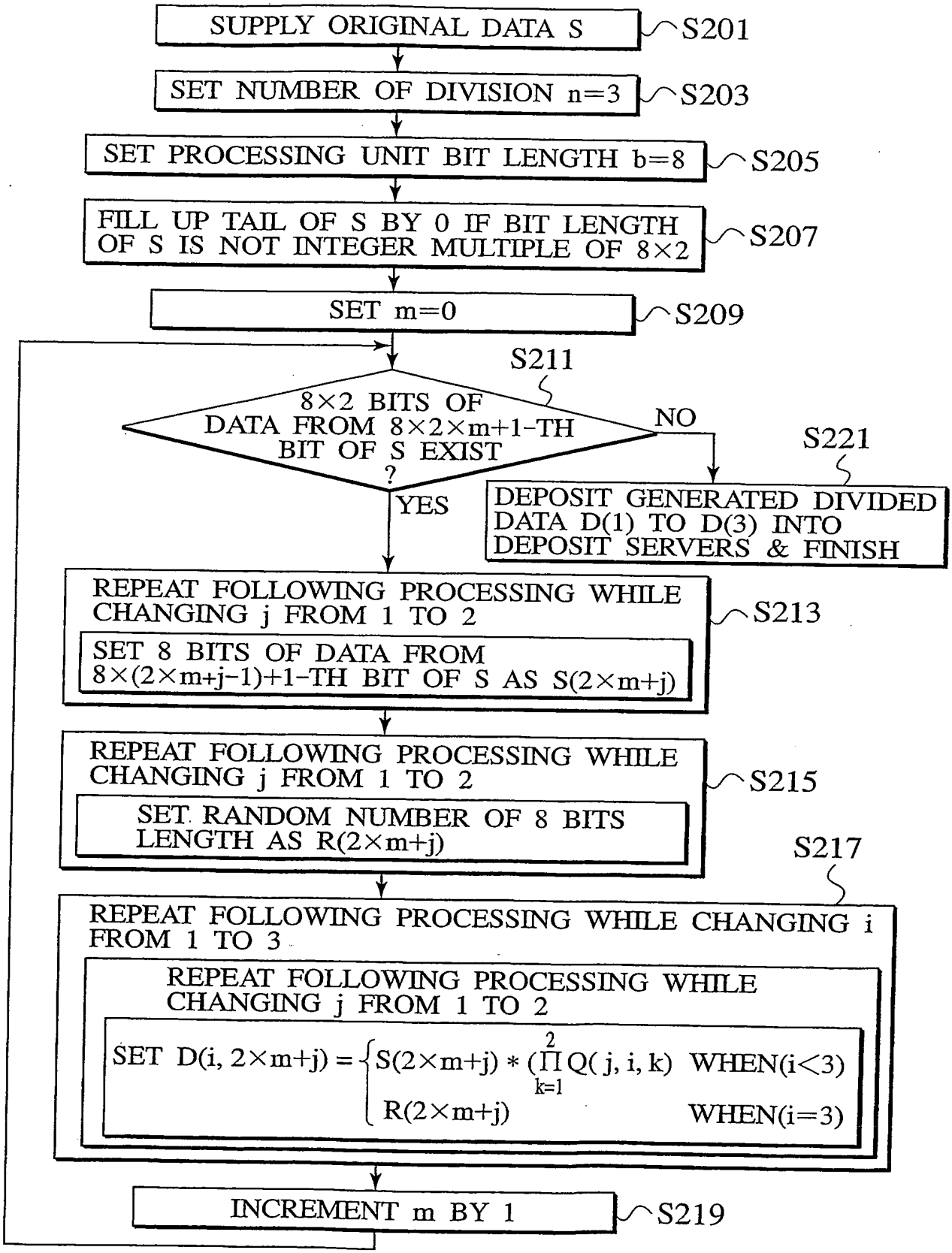
SUPPLY ORIGINAL DATA S
S201
SET NUMBER OF DIVISION n=3
S203
SET PROCESSING UNIT BIT LENGTH b=8
S205
FILL UP TAIL OF S BY 0 IF BIT LENGTH OF S IS NOT INTEGER MULTIPLE OF 8×2
S207
SET m=0
S209
S211
8×2 BITS OF DATA FROM 8×2×m+1-TH BIT OF S EXIST ?
NO
YES
S221
DEPOSIT GENERATED DIVIDED DATA D(1) TO D(3) INTO DEPOSIT SERVERS & FINISH
REPEAT FOLLOWING PROCESSING WHILE CHANGING j FROM 1 TO 2
SET 8 BITS OF DATA FROM 8×(2×m+j-1)+1-TH BIT OF S AS S(2×m+j)
S213
REPEAT FOLLOWING PROCESSING WHILE CHANGING j FROM 1 TO 2
SET RANDOM NUMBER OF 8 BITS LENGTH AS R(2×m+j)
S215
S217
REPEAT FOLLOWING PROCESSING WHILE CHANGING i FROM 1 TO 3
REPEAT FOLLOWING PROCESSING WHILE CHANGING j FROM 1 TO 2
SET D(i, 2×m+j) = S(2×m+j) * (∏ Q(j, i, k)) WHEN(i<3)
R(2×m+j) WHEN(i=3)
INCREMENT m BY 1
S219

FIG.3

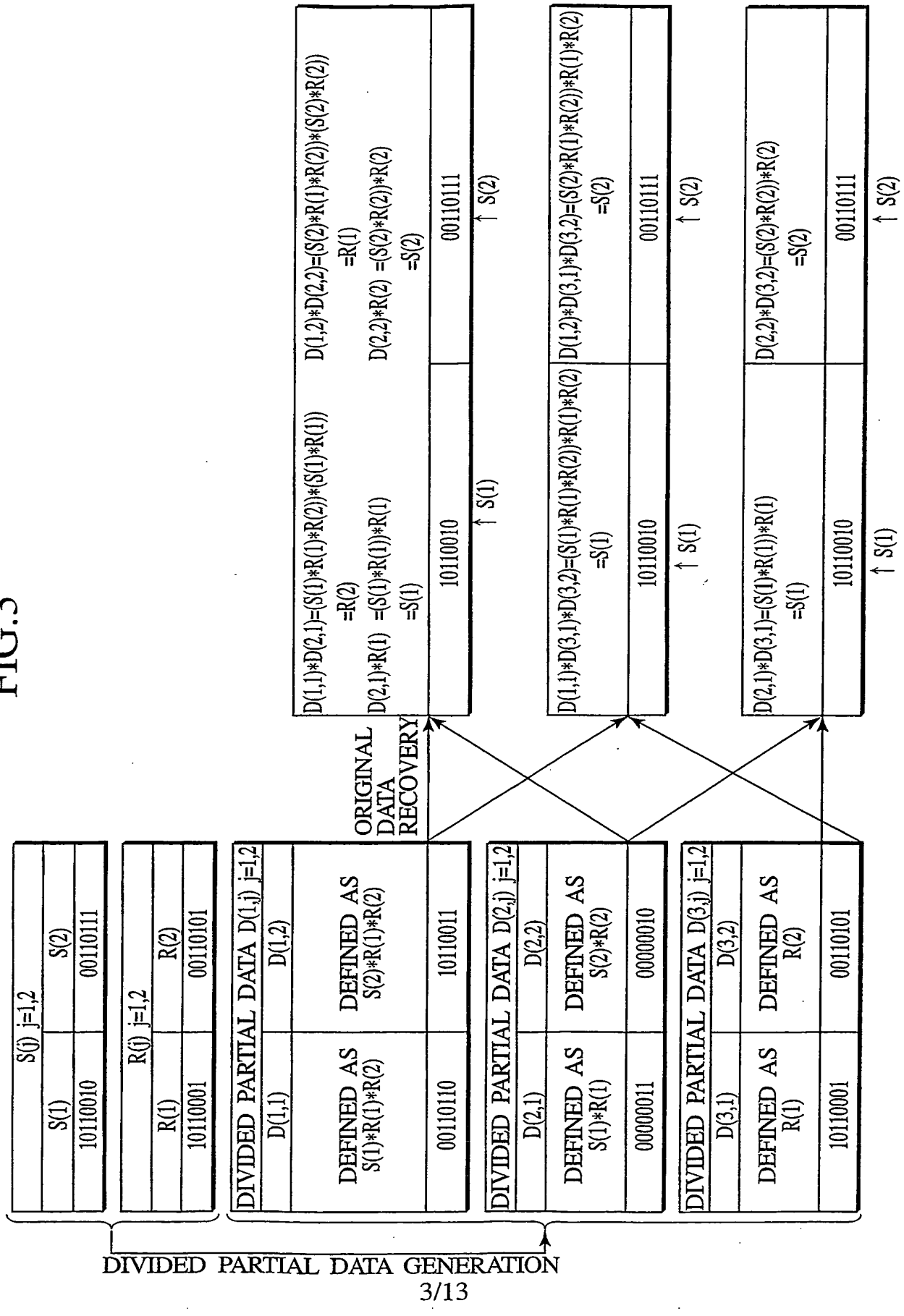
S(j) j=1,2
S(1)
S(2)
10110010
00110111
R(j) j=1,2
R(1)
R(2)
10110001
00110101
DIVIDED PARTIAL DATA D(1,j) j=1,2
D(1,1)
D(1,2)
DEFINED AS S(1)*R(1)*R(2)
DEFINED AS S(2)*R(1)*R(2)
00110110
10110011
DIVIDED PARTIAL DATA D(2,j) j=1,2
D(2,1)
D(2,2)
DEFINED AS S(1)*R(1)
DEFINED AS S(2)*R(2)
00000011
00000010
DIVIDED PARTIAL DATA D(3,j) j=1,2
D(3,1)
D(3,2)
DEFINED AS R(1)
DEFINED AS R(2)
10110001
00110101
DIVIDED PARTIAL DATA GENERATION
ORIGINAL DATA RECOVERY
D(1,1)*D(2,1)=(S(1)*R(1)*R(2))*(S(1)*R(1))
=R(2)
D(2,1)*R(1) =(S(1)*R(1))*R(1)
=S(1)
D(1,2)*D(2,2)=(S(2)*R(1)*R(2))*(S(2)*R(2))
=R(1)
D(2,2)*R(2) =(S(2)*R(2))*R(2)
=S(2)
10110010
00110111
↑ S(1)
↑ S(2)
D(1,1)*D(3,1)*D(3,2)=(S(1)*R(1)*R(2))*R(1)*R(2)
=S(1)
D(1,2)*D(3,1)*D(3,2)=(S(2)*R(1)*R(2))*R(1)*R(2)
=S(2)
10110010
00110111
↑ S(1)
↑ S(2)
D(2,1)*D(3,1)=(S(1)*R(1))*R(1)
=S(1)
D(2,2)*D(3,2)=(S(2)*R(2))*R(2)
=S(2)
10110010
00110111
↑ S(1)
↑ S(2)

FIG.4

DIVISION INTO THREE (n=3)
ORIGINAL DATA CAN BE RECOVERED FROM ANY TWO DIVIDED DATA

(m IS ARBITRARY INTEGER m>0)

VALUE OF j	1	2	···	j=2×m+1	j+1	···
ORIGINAL DATA S(j)	S(1)	S(2)	···	S(j)	S(j+1)	···
RANDOM NUMBER R(j)	R(1)	R(2)	···	R(j)	R(j+1)	···
DIVIDED PARTIAL DATA D(1, j)	S(1)*R(1)*R(2)	S(2)*R(1)*R(2)	···	S(j)*R(j)*R(j+1)	S(j+1)*R(j)*R(j+1)	···
DIVIDED PARTIAL DATA D(2, j)	S(1)*R(1)	S(2) *R(2)	···	S(j)*R(j)	S(j+1) *R(j+1)	···
DIVIDED PARTIAL DATA D(3, j)	R(1)	R(2)	···	R(j)	R(j+1)	···

→CONTINUED TO TAIL
OF ORIGINAL DATA S

PCT/JP2003/016389

FIG.5

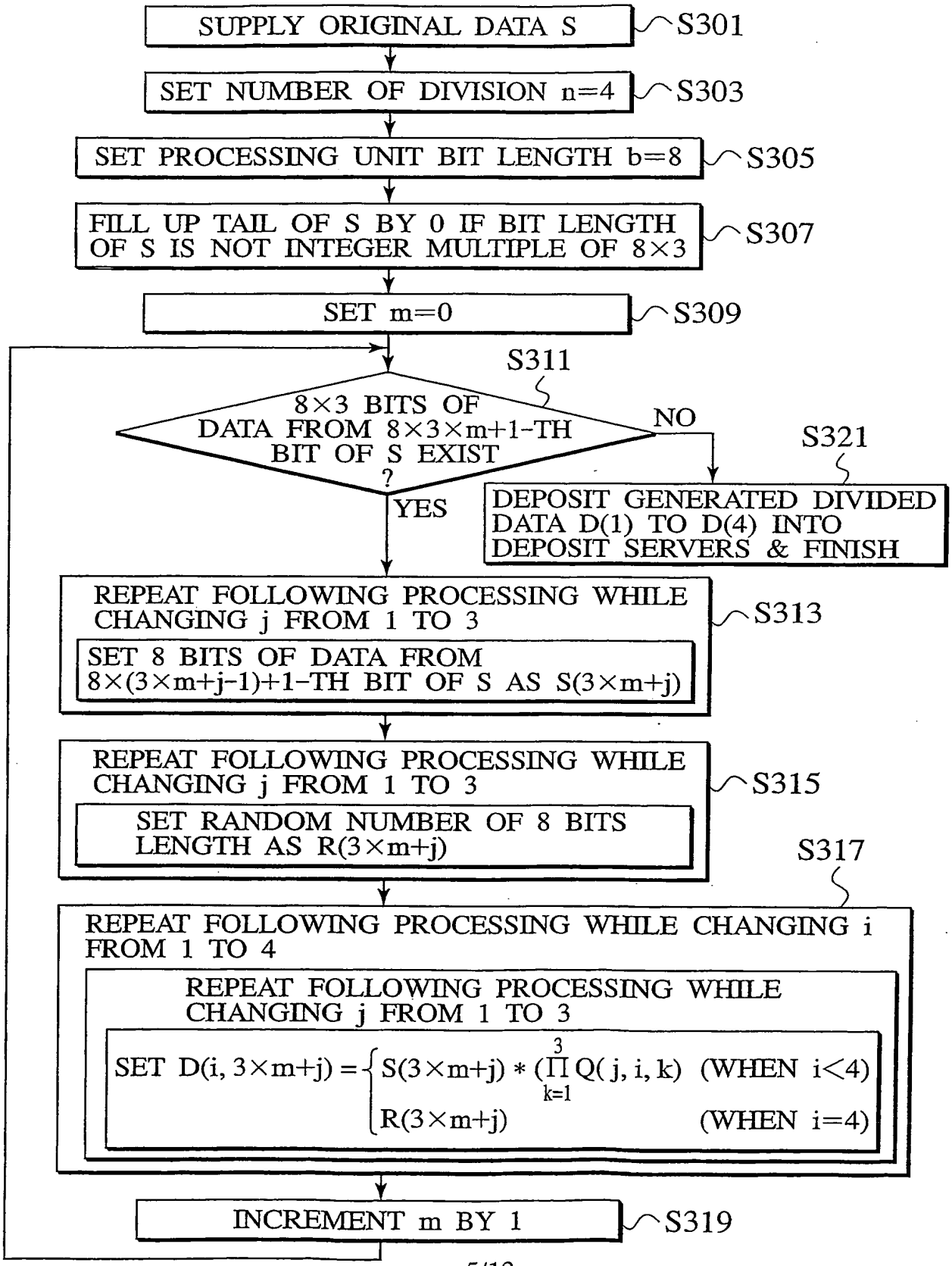
SUPPLY ORIGINAL DATA S
S301
SET NUMBER OF DIVISION n=4
S303
SET PROCESSING UNIT BIT LENGTH b=8
S305
FILL UP TAIL OF S BY 0 IF BIT LENGTH OF S IS NOT INTEGER MULTIPLE OF 8×3
S307
SET m=0
S309
S311
8×3 BITS OF DATA FROM 8×3×m+1-TH BIT OF S EXIST ?
NO
YES
S321
DEPOSIT GENERATED DIVIDED DATA D(1) TO D(4) INTO DEPOSIT SERVERS & FINISH
REPEAT FOLLOWING PROCESSING WHILE CHANGING j FROM 1 TO 3
SET 8 BITS OF DATA FROM 8×(3×m+j-1)+1-TH BIT OF S AS S(3×m+j)
S313
REPEAT FOLLOWING PROCESSING WHILE CHANGING j FROM 1 TO 3
SET RANDOM NUMBER OF 8 BITS LENGTH AS R(3×m+j)
S315
S317
REPEAT FOLLOWING PROCESSING WHILE CHANGING i FROM 1 TO 4
REPEAT FOLLOWING PROCESSING WHILE CHANGING j FROM 1 TO 3
SET D(i, 3×m+j) = { S(3×m+j) * (∏ k=1 to 3 Q(j, i, k)) (WHEN i<4)
R(3×m+j) (WHEN i=4)
INCREMENT m BY 1
S319

FIG.6

DIVISION INTO FOUR (n=4)

ORIGINAL DATA CAN BE RECOVERED FROM
ANY THREE DIVIDED DATA(OR TWO DIVIDED DATA IN SOME CASES)

VALUE OF j	1	2	3	···
ORIGINAL DATA S(j)	S(1)	S(2)	S(3)	···
RANDOM NUMBER R(j)	R(1)	R(2)	R(3)	···
DIVIDED PARTIAL DATA D(1, j)	S(1)*R(1)*R(2)*(R3)	S(2)*R(1)*R(2)*(R3)	S(3)*R(1)*R(2)*(R3)	···
DIVIDED PARTIAL DATA D(2, j)	S(1)*R(1)*R(2)	S(2) *R(2)*(R3)	S(3)*R(1) *(R3)	···
DIVIDED PARTIAL DATA D(3, j)	S(1)*R(1)	S(2) *R(2)	S(3) *R(3)	···
DIVIDED PARTIAL DATA D(4, j)	R(1)	R(2)	R(3)	···

(A)

(m IS ARBITRARY INTEGER m>0)

···	j=3×m+1	j+1	j+2	···
···	S(j)	S(j+1)	S(j+2)	···
···	R(j)	R(j+1)	R(j+2)	···
···	S(j)*R(j)*R(j+1)*R(j+2)	S(j+1)*R(j)*R(j+1)*R(j+2)	S(j+2)*R(j)*R(j+1)*R(j+2)	···
···	S(j)*R(j)*R(j+1)	S(j+1) *R(j+1)*R(j+2)	S(j+2)*R(j) *R(j+2)	···
···	S(j)*R(j)	S(j+1) *R(j+1)	S(j+2) *R(j+2)	···
···	R(j)	R(j+1)	R(j+2)	···

(A)

→CONTINUED TO TAIL OF ORIGINAL DATA S

10/539846
PCT/JP2003/016389

FIG.7

DIVISION INTO FIVE (n=5)

ORIGINAL DATA CAN BE RECOVERED FROM
ANY THREE DIVIDED DATA(OR TWO DIVIDED DATA IN SOME CASES)

VALUE OF j	1	2	3	4	···
ORIGINAL DATA S(j)	S(1)	S(2)	S(3)	S(4)	···
RANDOM NUMBER (j)	R(1)	R(2)	R(3)	R(4)	···
DIVIDED PARTIAL DATA D(1, j)	S(1)*R(1)*R(2)*(R3)*(R4)	S(2)*R(1)*R(2)*(R3)*(R4)	S(3)*R(1)*R(2)*(R3)*(R4)	S(4)*R(1)*R(2)*(R3)*(R4)	···
DIVIDED PARTIAL DATA D(2, j)	S(1)*R(1)*R(2)*(R3)	S(2) *R(2)*(R3)*(R4)	S(3)*R(1) *(R3)*(R4)	S(4)*R(1)*R(2) *(R4)	···
DIVIDED PARTIAL DATA D(3, j)	S(1)*R(1)*R(2)	S(2) *R(2)*(R3)	S(3) *R(3)*(R4)	S(4)*R(1) *(R4)	···
DIVIDED PARTIAL DATA D(4, j)	S(1)*R(1)	S(2) *R(2)	S(3) *R(3)	S(4) *(R4)	···
DIVIDED PARTIAL DATA D(5, j)	R(1)	R(2)	R(3)	R(4)	···

B

(m IS ARBITRARY INTEGER m>0)

B

···	j=4×m+1	j+1	j+2	j+3	···
···	S(j)	S(j+1)	S(j+2)	S(j+3)	···
···	R(j)	R(j+1)	R(j+2)	R(j+3)	···
···	S(j)*R(j)*R(j+1)*R(j+2)*R(j+3)	S(j+1)*R(j)*R(j+1)*R(j+2)*R(j+3)	S(j+2)*R(j)*R(j+1)*R(j+2)*R(j+3)	S(j+3)*R(j)*R(j+1)*R(j+2)*R(j+3)	···
···	S(j)*R(j)*R(j+1)*R(j+2)	S(j+1) *R(j+1)*R(j+2)*R(j+3)	S(j+2)*R(j) *R(j+2)*R(j+3)	S(j+3)*R(j)*R(j+1) *R(j+3)	···
···	S(j)*R(j)*R(j+1)	S(j+1) *R(j+1)*R(j+2)	S(j+2) *R(j+2)*R(j+3)	S(j+3)*R(j) *R(j+3)	···
···	S(j)*R(j)	S(j+1) *R(j+1)	S(j+2) *R(j+2)	S(j+3) *R(j+3)	···
···	R(j)	R(j+1)	R(j+2)	R(j+3)	···

→CONTINUED TO TAIL
OF ORIGINAL DATA S

FIG.8

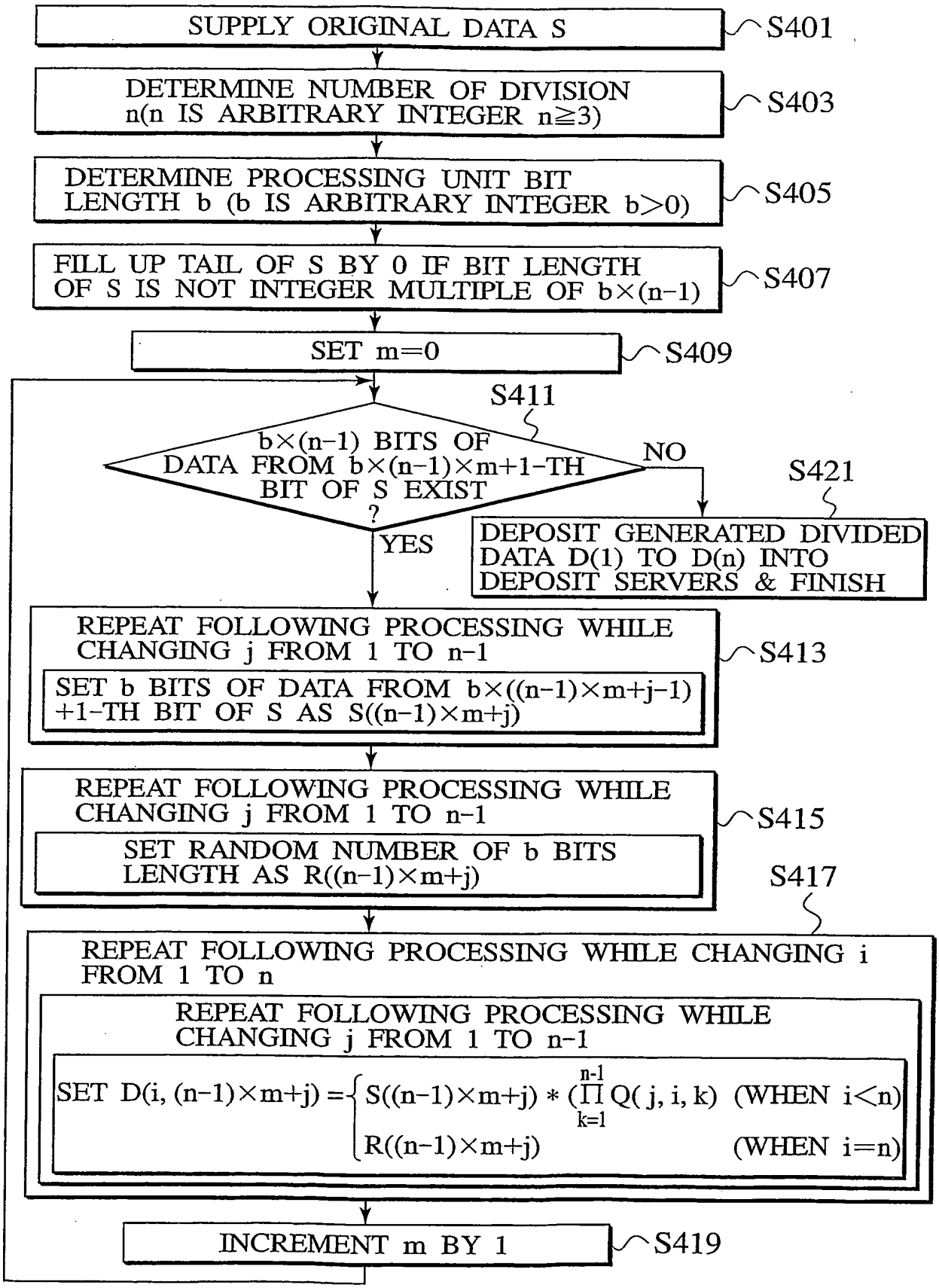
SUPPLY ORIGINAL DATA S
S401
DETERMINE NUMBER OF DIVISION n(n IS ARBITRARY INTEGER n≧3)
S403
DETERMINE PROCESSING UNIT BIT LENGTH b (b IS ARBITRARY INTEGER b>0)
S405
FILL UP TAIL OF S BY 0 IF BIT LENGTH OF S IS NOT INTEGER MULTIPLE OF b×(n-1)
S407
SET m=0
S409
S411
b×(n-1) BITS OF DATA FROM b×(n-1)×m+1-TH BIT OF S EXIST ?
NO
YES
S421
DEPOSIT GENERATED DIVIDED DATA D(1) TO D(n) INTO DEPOSIT SERVERS & FINISH
REPEAT FOLLOWING PROCESSING WHILE CHANGING j FROM 1 TO n-1
SET b BITS OF DATA FROM b×((n-1)×m+j-1)+1-TH BIT OF S AS S((n-1)×m+j)
S413
REPEAT FOLLOWING PROCESSING WHILE CHANGING j FROM 1 TO n-1
SET RANDOM NUMBER OF b BITS LENGTH AS R((n-1)×m+j)
S415
S417
REPEAT FOLLOWING PROCESSING WHILE CHANGING i FROM 1 TO n
REPEAT FOLLOWING PROCESSING WHILE CHANGING j FROM 1 TO n-1
SET D(i, (n-1)×m+j) = { S((n-1)×m+j) * (∏ k=1 to n-1 Q(j, i, k)) (WHEN i<n)
R((n-1)×m+j) (WHEN i=n)
INCREMENT m BY 1
S419

FIG.9

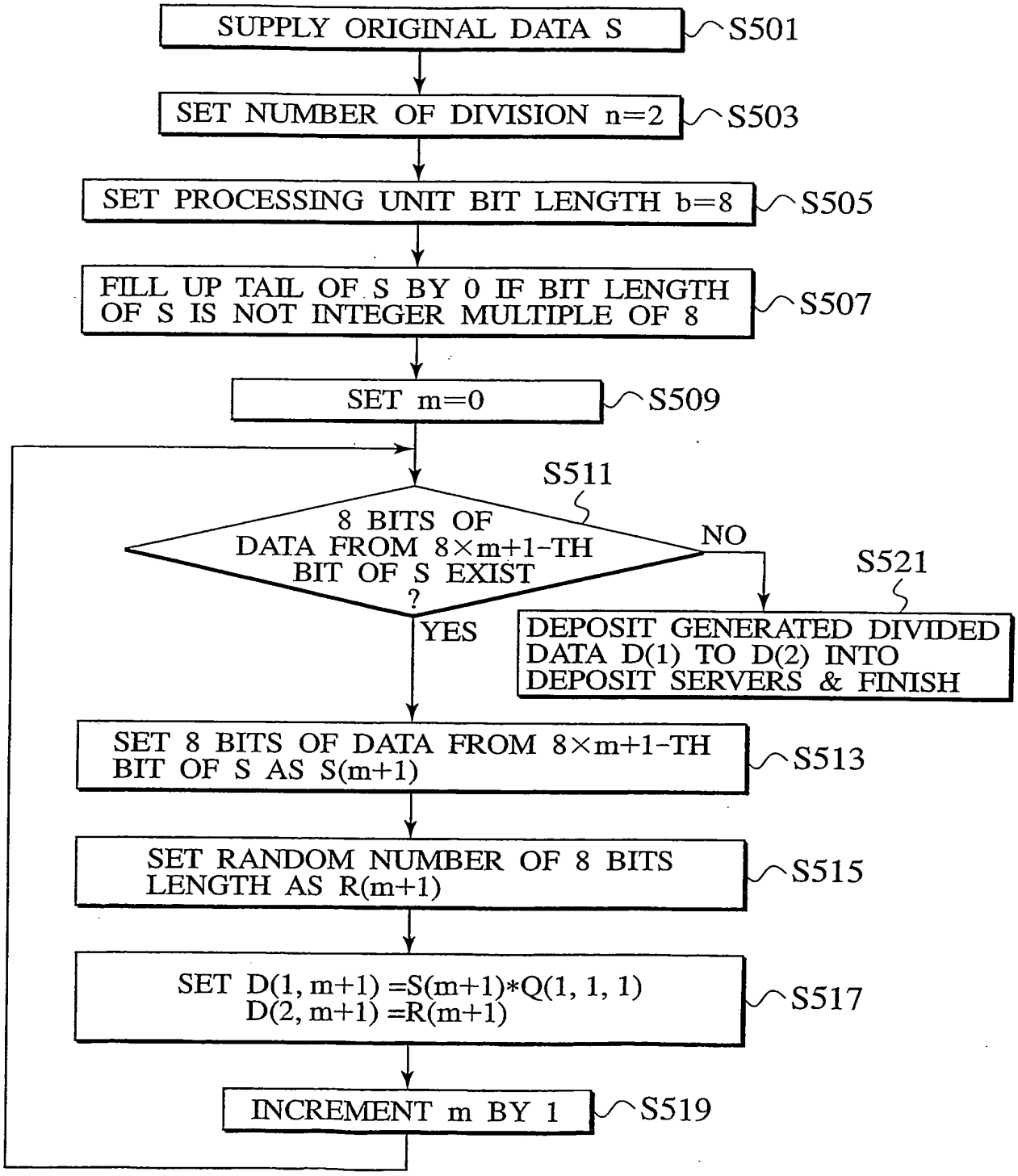
SUPPLY ORIGINAL DATA S
S501
SET NUMBER OF DIVISION n=2
S503
SET PROCESSING UNIT BIT LENGTH b=8
S505
FILL UP TAIL OF S BY 0 IF BIT LENGTH OF S IS NOT INTEGER MULTIPLE OF 8
S507
SET m=0
S509
S511
8 BITS OF DATA FROM 8×m+1-TH BIT OF S EXIST ?
NO
YES
S521
DEPOSIT GENERATED DIVIDED DATA D(1) TO D(2) INTO DEPOSIT SERVERS & FINISH
SET 8 BITS OF DATA FROM 8×m+1-TH BIT OF S AS S(m+1)
S513
SET RANDOM NUMBER OF 8 BITS LENGTH AS R(m+1)
S515
SET D(1, m+1) =S(m+1)*Q(1, 1, 1)
D(2, m+1) =R(m+1)
S517
INCREMENT m BY 1
S519

FIG.10

DIVISION INTO THREE (n=3)
ORIGINAL DATA CAN BE RECOVERED FROM ANY TWO DIVIDED DATA

(m IS ARBITRARY INTEGER m>0)

VALUE OF j	1	2	···	j=2×m+1	j+1	···
ORIGINAL DATA S(j)	S(1)	S(2)	···	S(j)	S(j+1)	···
RANDOM NUMBER R(j)	R(1)	R(2)	···	R(j)	R(j+1)	···
DIVIDED PARTIAL DATA D(1, j)	S(1)*R(1)*R(2)	S(2) *R(2)	···	S(j)*R(j)*R(j+1)	S(j+1) *R(j+1)	···
DIVIDED PARTIAL DATA D(2, j)	S(1)*R(1)	S(2)*R(1)*R(2)	···	S(j)*R(j)	S(j+1)*R(j)*R(j+1)	···
DIVIDED PARTIAL DATA D(3, j)	R(1)	R(2)	···	R(j)	R(j+1)	···

→CONTINUED TO TAIL OF ORIGINAL DATA S

10/539846
PCT/JP2003/016389

FIG.11

DIVISION INTO THREE (n=3)
ORIGINAL DATA CAN BE RECOVERED FROM ANY TWO DIVIDED DATA

(m IS ARBITRARY INTEGER m>0)

VALUE OF j	1	2	···	j=2×m+1	j+1	···
ORIGINAL DATA S(j)	S(1)	S(2)	···	S(j)	S(j+1)	···
RANDOM NUMBER R(j)	R(1)	R(2)	···	R(j)	R(j+1)	···
DIVIDED PARTIAL DATA D(1, j)	S(1) *R(2)	S(2)*R(1)*R(2)	···	S(j) *R(j+1)	S(j+1)*R(j)*R(j+1)	···
DIVIDED PARTIAL DATA D(2, j)	S(1)*R(1)	S(2) *R(2)	···	S(j)*R(j)	S(j+1) *R(j+1)	···
DIVIDED PARTIAL DATA D(3, j)	R(1)	R(2)	···	R(j)	R(j+1)	···

→CONTINUED TO TAIL
OF ORIGINAL DATA S

PCT/JP2003/016389

FIG.12

DIVISION INTO FOUR (n=4)

ORIGINAL DATA CAN BE RECOVERED FROM ANY THREE DIVIDED DATA(OR TWO DIVIDED DATA IN SOME CASES)

VALUE OF j	1	2	3	···
ORIGINAL DATA S(j)	S(1)	S(2)	S(3)	···
RANDOM NUMBER R(j)	R(1)	R(2)	R(3)	···
DIVIDED PARTIAL DATA D(1, j)	S(1) *R(2)*(R3)	S(2)*R(1)*R(2)*(R3)	S(3)*R(1)*R(2)*(R3)	···
DIVIDED PARTIAL DATA D(2, j)	S(1) *R(2)	S(2) *R(2)*(R3)	S(3)*R(1) *(R3)	···
DIVIDED PARTIAL DATA D(3, j)	S(1)*R(1)	S(2) *R(2)	S(3) *R(3)	···
DIVIDED PARTIAL DATA D(4, j)	R(1)	R(2)	R(3)	···

(C)

(m IS ARBITRARY INTEGER m>0)

(C)

···	j=3×m+1	j+1	j+2	···
···	S(j)	S(j+1)	S(j+2)	···
···	R(j)	R(j+1)	R(j+2)	···
···	S(j) *R(j+1)*R(j+2)	S(j+1)*R(j)*R(j+1)*R(j+2)	S(j+2)*R(j)*R(j+1)*R(j+2)	···
···	S(j) *R(j+1)	S(j+1) *R(j+1)*R(j+2)	S(j+2)*R(j) *R(j+2)	···
···	S(j)*R(j)	S(j+1) *R(j+1)	S(j+2) *R(j+2)	···
···	R(j)	R(j+1)	R(j+2)	···

→CONTINUED TO TAIL OF ORIGINAL DATA S

10/539846
PCT/JP2003/016389

FIG.13

DIVISION INTO FIVE (n=5)

ORIGINAL DATA CAN BE RECOVERED FROM
ANY THREE DIVIDED DATA(OR TWO DIVIDED DATA IN SOME CASES)

VALUE OF j	1	2	3	4	···
ORIGINAL DATA S(j)	S(1)	S(2)	S(3)	S(4)	···
RANDOM NUMBER (j)	R(1)	R(2)	R(3)	R(4)	···
DIVIDED PARTIAL DATA D(1, j)	S(1) *R(2)*(R3)*(R4)	S(2)*R(1)*R(2)*(R3)*(R4)	S(3)*R(1)*R(2)*(R3)*(R4)	S(4)*R(1)*R(2)*(R3)*(R4)	···
DIVIDED PARTIAL DATA D(2, j)	S(1) *R(2)*(R3)	S(2) *R(2)*(R3)*(R4)	S(3)*R(1) *(R3)*(R4)	S(4)*R(1)*R(2) *(R4)	···
DIVIDED PARTIAL DATA D(3, j)	S(1) *R(2)	S(2) *R(2)*(R3)	S(3) *R(3)*(R4)	S(4)*R(1) *(R4)	···
DIVIDED PARTIAL DATA D(4, j)	S(1)*R(1)	S(2) *R(2)	S(3) *R(3)	S(4) *(R4)	···
DIVIDED PARTIAL DATA D(5, j)	R(1)	R(2)	R(3)	R(4)	···

(D)

(m IS ARBITRARY INTEGER m>0)

···	j=4×m+1	j+1	j+2	j+3	···
···	S(j)	S(j+1)	S(j+2)	S(j+3)	···
···	R(j)	R(j+1)	R(j+2)	R(j+3)	···
···	S(j) *R(j+1)*R(j+2)*R(j+3)	S(j+1)*R(j)*R(j+1)*R(j+2)*R(j+3)	S(j+2)*R(j)*R(j+1)*R(j+2)*R(j+3)	S(j+3)*R(j)*R(j+1)*R(j+2)*R(j+3)	···
···	S(j) *R(j+1)*R(j+2)	S(j+1) *R(j+1)*R(j+2)*R(j+3)	S(j+2)*R(j) *R(j+2)*R(j+3)	S(j+3)*R(j)*R(j+1) *R(j+3)	···
···	S(j) *R(j+1)	S(j+1) *R(j+1)*R(j+2)	S(j+2) *R(j+2)*R(j+3)	S(j+3)*R(j) *R(j+3)	···
···	S(j)*R(j)	S(j+1) *R(j+1)	S(j+2) *R(j+2)	S(j+3) *R(j+3)	···
···	R(j)	R(j+1)	R(j+2)	R(j+3)	···

(D)

→CONTINUED TO TAIL
OF ORIGINAL DATA S